Improving Consistency-Based Semi-Supervised Learning with Weight Averaging

Ben Athiwaratkun, Marc Finzi, Pavel Izmailov, Andrew Gordon Wilson
{pa338, maf388, pi49, andrew}@cornell.edu
Cornell University

Abstract

Recent advances in deep unsupervised learning have renewed interest in semi-supervised methods, which can learn from both labeled and unlabeled data. Presently the most successful approaches to semi-supervised learning are based on *consistency regularization*, whereby a model is trained to be robust to small perturbations of its inputs and parameters. We show that consistency regularization leads to flatter but narrower optima. We also show that the test error surface for these methods is approximately convex in regions of weight space traversed by SGD. Inspired by these observations, we propose to train consistency based semi-supervised models with *stochastic weight averaging* (SWA), a recent method which averages weights along the trajectory of SGD. We also develop *fast-SWA*, which further accelerates convergence by averaging multiple points within each cycle of a cyclical learning rate schedule. With fast-SWA we achieve the best known semi-supervised results on CIFAR-10 and CIFAR-100 over many different numbers of observed training labels. For example, we achieve 95.0% accuracy on CIFAR-10 with only 4000 labels, compared to the previous best result in the literature of 93.7%. We also improve the best known accuracy for domain adaptation from CIFAR-10 to STL from 80% to 83%. Finally, we show that with fast-SWA the simple Π model becomes state-of-the-art for large labeled settings.

1 Introduction

Recent advances in deep unsupervised learning, such as generative adversarial networks (GANs) [7], have led to an explosion of interest in semi-supervised learning. Semi-supervised methods make use of both unlabeled and labeled training data to improve performance over purely supervised methods. Semi-supervised learning is particularly valuable in applications such as medical imaging, where labeled data may be scarce and expensive [20].

Semi-supervised methods based on GANs have achieved promising results [e.g., 24, 22, 3, 13]. The bottleneck in applying GANs to semi-supervised learning is in training at scale, and creating a generator network that matches the power of a discriminator network, which is presently difficult for state-of-the-art deep residual discriminators [24, 3].

Currently the best semi-supervised results are obtained by *consistency-enforcing* approaches [16, 28, 19]. These methods use unlabeled data to stabilize their predictions under input or weight perturbations. Consistency-enforcing methods can be used at scale with state-of-the-art architectures. For example, the recent Mean Teacher [28] model has been used with the Shake-Shake [6] architecture for the best semi-supervised performance on the consequential CIFAR benchmarks.

In this paper, we analyze the geometry of the training objective for the popular Π [16] and Mean Teacher [28] consistency models. Inspired by our findings, we propose a training procedure based on stochastic weight averaging (SWA) [10]. On a thorough empirical study we show this procedure achieves the best known semi-supervised results on consequential benchmarks. In particular:

- We show that the Π model is implicitly regularizing the norm of the Jacobian of the network outputs with respect to its weights averaged over the labeled and unlabeled data, which encourages flatter solutions. Flat optima have been associated with improved generalization performance [see e.g. 25, 11, 10].
- We empirically find that the Π and MT models find flatter but narrower solutions in the sense that under small perturbations of weights test error stays almost constant, but for larger perturbations it quickly deteriorates.
- We show empirically that the test error is approximately convex in weights in the region of weight space traversed by the last epochs of SGD training for the Mean Teacher and Π models, and supervised training. To demonstrate this convexity, we show that Jensen's inequality holds for random convex combinations of networks from later epochs of the SGD trajectory.
- Motivated by these observations, we propose to train consistency based semi-supervised models using stochastic weight averaging (SWA) [10]. Moreover, we propose *fast-SWA* which improves convergence by averaging multiple weights traversed by SGD on each cycle of a cyclical learning rate schedule. Although our focus is on semi-supervised learning, we note that fast-SWA could also be applied to supervised problems.
- Applying fast-SWA to the Π and Mean Teacher models we improve the best reported results on CIFAR-10 for $1k$, $2k$, $4k$ and $10k$ labeled examples, as well as on CIFAR-100 with $10k$ labeled examples. For example, we obtain 95% accuracy on CIFAR-10 with only $4k$ labels, improving the best result reported in the literature [28] by 1.28%, which is particularly significant at this high accuracy level. We also apply fast-SWA to a state-of-the-art domain adaptation technique [5] closely related to the Mean Teacher model and improve the best results on domain adaptation from CIFAR-10 to STL from 80.1% to 82.9% accuracy. We also find that fast-SWA improves the relatively simple Π model to achieve state-of-the-art-results in large labeled regimes (e.g. CIFAR-10 with $4k$ and 10k labels).
- We release our code for semi-supervised learning with fast-SWA at `https://github.com/benathi/fastswa-semi-sup`

2 Consistency Based Models

We briefly review semi-supervised learning with consistency-based models. This class of models encourages predictions to stay similar under small perturbations of inputs or network parameters. For instance, two different translations of the same image should result in similar predicted probabilities. The consistency of a model (student) can be measured against its own predictions (e.g. Π model) or predictions of a different teacher network (e.g. Mean Teacher model). In both cases we will say a *student* measures consistency against a *teacher* network, whether it be a network with the same or different architecture.

Consistency Loss In the semi-supervised setting, we have access to labeled data $\mathcal{D}_L = \{(x_i^L, y_i^L)\}_{i=1}^{N_L}$, and unlabeled data $\mathcal{D}_U = \{x_i^U\}_{i=1}^{N_U}$.

Given two perturbed inputs x', x'' of x and the perturbed weights w'_f and w'_g, the consistency loss penalizes the difference between the *student*'s predicted probablities $f(x'; w'_f)$ and the *teacher*'s $g(x''; w'_g)$. This loss is typically the Mean Squared Error or KL divergence:

$$\ell^{\text{MSE}}_{\text{cons}}(w_f, x) = \|f(x'; w'_f) - g(x'', w'_g)\|^2 \text{ or } \ell^{\text{KL}}_{\text{cons}}(w_f, x) = \text{KL}(f(x'; w'_f)||g(x'', w'_g))\,. \quad (1)$$

The total loss used to train the model can be written as

$$L(w_f) = \underbrace{\sum_{(x,y)\in\mathcal{D}_L} \ell_{\text{CE}}(w_f, x, y)}_{L_{\text{CE}}} + \lambda \underbrace{\sum_{x\in\mathcal{D}_L\cup\mathcal{D}_U} \ell_{\text{cons}}(w_f, x)}_{L_{\text{cons}}}, \quad (2)$$

where for classification L_{CE} is the cross entropy between the model predictions and supervised training labels. The parameter $\lambda > 0$ controls the relative importance of the consistency term in the overall loss.

Π Model The Π model, introduced in Laine and Aila [16] and Sajjadi et al. [23], uses the student model f as its own teacher. The data (input) perturbations include random translations, crops, flips and additive Gaussian noise. Binary dropout [27] is used for weight perturbation.

Mean Teacher Model The Mean Teacher model (MT) proposed in Tarvainen and Valpola [28] uses the same data and weight perturbations as the Π model; however, the teacher weights w^g are the exponential moving average (EMA) of the student weights w^f: $w^k_g = \alpha \cdot w^{k-1}_g + (1-\alpha) \cdot w^k_f$. The decay rate α is usually set between 0.9 and 0.999. The Mean Teacher model has the best known results on the CIFAR-10 semi-supervised learning benchmark.

Other Consistency-Based Models Temporal Ensembling (TE) [16] uses an exponential moving average of the student outputs as the teacher outputs in the consistency term for training. Another approach, Virtual Adversarial Training (VAT) [19], penalizes the consistency between the original data inputs and the data perturbed in an adversarial direction $x' = x + \epsilon r_{\text{adv}}$, where $r_{\text{adv}} = \arg\max_{r:\|r\|=1} \text{KL}[f(x, w)\|f(x + \xi r, w)]$.

3 Analysis of Consistency-Enforcing Methods

We analyze the properties of solutions found by the popular Π and Mean Teacher consistency-based semi-supervised methods. In Section 3.1 we empirically demonstrate that these methods converge to solutions that are substantially flatter under small weight perturbations than those of supervised training. In Section 3.2 we show that SGD traverses convex regions of the test error surface for these methods. In Appendix A.3 and A.4 we theoretically analyze how consistency losses encourage flatter optima.

3.1 Solution Geometry of Consistency-Enforcing Methods

The Π and Mean Teacher models enforce consistency of predictions under small weight and data perturbations (see Section 2 for details). Intuitively, networks with predictions consistent under small weight perturbations lie in a flat region of test error. Indeed, if under small weight perturbations the predictions of the network stay roughly constant, the test error should also stay approximately constant. We formally analyze the connection between

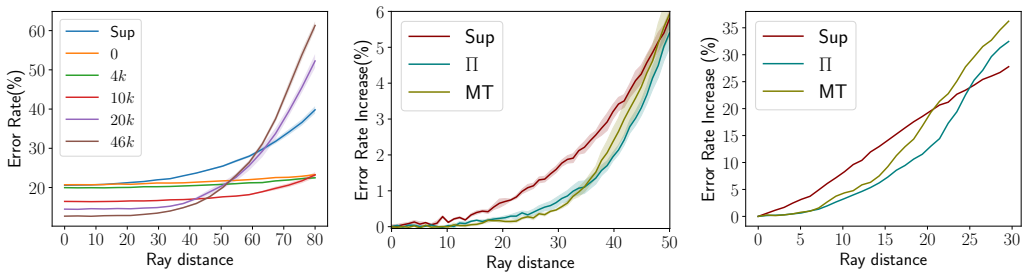


Figure 1: All plots are on CIFAR-10, using a 13-layer CNN. Shaded regions correspond to one standard error. **Left:** Test error as a function of distance along random rays for the Π model with 0, $4k$, $10k$, $20k$ or $46k$ unlabeled data points, and standard fully supervised training which uses only the cross entropy loss. All methods use $4k$ labeled examples. **Middle** and **Right**: Test error as a function of distance along random (middle) and adversarial (right) rays for standard supervised training, Π model and Mean Teacher solutions, using $4k$ labeled examples and $46k$ unlabeled examples.

flatness and consistency under small weight perturbations in Section A.3 and show that a simplified version of Π model indeed encourages flat solutions. In the present section we examine whether the full Π and closely related Mean Teacher models do indeed converge to flatter optima in practice.

In Figure 1 (left) we visualize how test accuracy changes along random rays starting at the solution found by the Π model trained with $4k$ labeled points and different amounts of additional unlabeled data (0, $4k$, $10k$, $20k$ and $46k$) on CIFAR-10. With 0 unlabeled points, the Π model is technically doing fully supervised learning, but trained using both cross entropy and consistency loss terms in Eq. (2). We also compare to standard supervised training (Sup), corresponding to training only with the L_{CE} term in Eq. (2).

Let $\text{Err}(w)$ be the test error rate for the network with weights w. For this plot, we sample 5 random vectors d from the unit sphere, and calculate average error $\text{Err}(w + sd)$ along the ray for $s \in [0, 80]$. We refer to the plot of the error as a function of s as an *error curve*.

We observe that increasing the number of unlabeled examples improves the generalization performance of the Π model. We also see that the Π model produces solutions with very flat error curves for short distances s along the ray. However, we also see that for large amounts of unlabeled data and large sizes s of weight perturbations the accuracy of the Π model deteriorates rapidly, and the slope of the error curve increases with the number of unlabeled points. For large perturbations, the solutions of the Π model have *increased* error with an increased number of unlabeled data. In other words, these optima are locally flat, but also *narrow*, and become narrower as we increase the amount of unlabeled data. We also see in Figure 1 (middle) that the solutions found using the Π model and Mean Teacher are flatter but narrower than the standard supervised model trained only using L_{CE} in Eq. (2). Here we use $4k$ labels and align each of the curves at a distance of $s = 0$, to directly compare their shape.

To further analyze the geometry of the Π and Mean Teacher solutions we evaluate the error along adversarial rays in Figure 1 (right). Adversarial rays constitute the directions of fastest ascent of test error and correspond to the normalized gradient of the test cross entropy loss, $d_{\text{adv}} = \frac{\nabla L_{\text{CE}}}{\|\nabla L_{\text{CE}}\|}$. As in the middle panel, we align the error curves at a distance of zero, and see that along adversarial rays the Π and Mean Teacher solutions are again flatter but narrower than the supervised solutions.

3.2 Approximate Convexity of Test Error Surface

The loss surfaces for training deep neural networks are highly non-convex. However, along the trajectories traversed by SGD, the loss surface may be approximately convex [8]. In this section we analyze the convexity of the test error near the semi-supervised Π and Mean Teacher solutions, and standard supervised-only solutions. Standard supervised methods only use the L_{CE} loss in Eq. (2). We show that the test accuracy is approximately convex in the region of weight space traversed by later epochs of training for all three methods. The convexity of test error has high practical importance. If the test error is approximately convex, Jensen's inequality implies that averaging the weights of multiple networks with roughly the same performance will lead to a new network with performance at least as good as the average performance of the networks. If the test error is non-convex in the weights of the networks, this result may not hold. Consider Figure 2 (left), illustrating the difference between combining weights for convex and non-convex loss functions. For all w_1 and w_2, the convex function (green) evaluated on any convex combination of w_1 and w_2, which corresponds to using a network with combined weights, is below the dotted green line, which corresponds to combining network outputs and averaging performance. But for the non-convex function (blue) there exists convex combinations of w_1 and w_2 above the blue dotted line.

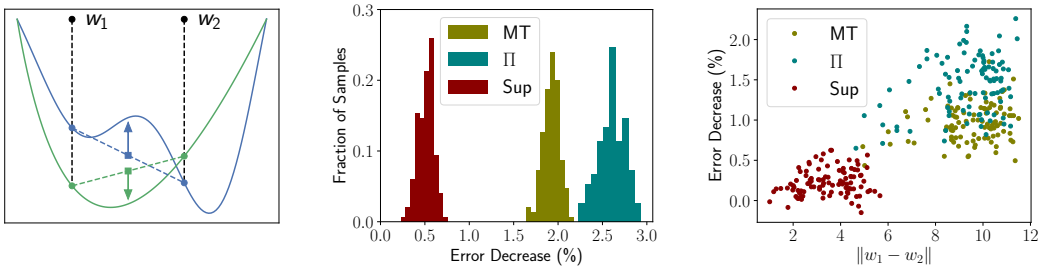
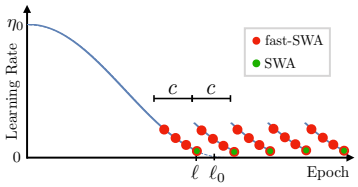


Figure 2: CIFAR-10 with $4k$ labeled examples (and $46k$ unlabeled examples), 13-layer CNN. **Left**: Illustration of a convex and non-convex function and Jensen's inequality. **Middle:** Histogram of the increase in accuracy of averaging random convex combinations of points from SGD trajectory compared to the corresponding convex combinations of their accuracies for Π model, Mean Teacher and supervised-only training. **Right:** Scatter plot of the increase in accuracy of averaging random pairs of points from the SGD trajectory compared to the average of their accuracies against distance between these points for Π model, Mean Teacher and supervised-only training.

In particular, if the test error as a function of network weights w, $\text{Err}(w)$, is convex, then for any convex combination of weights w_i, $i = 1, \ldots, n$ with coefficients $\alpha_1, \alpha_2, \ldots, \alpha_n$, where $\sum_i \alpha_i = 1$, $\alpha_i \geq 0$ for all i, the difference $C(\alpha) \equiv \sum_i \alpha_i \text{Err}(w_i) - \text{Err}\left(\sum_i \alpha_i w_i\right) \geq 0$, by Jensen's inequality.

We now use C to evaluate the benefit of weight averaging along the trajectory explored by SGD for the Mean Teacher, Π model, and fully-supervised training methods. We train a 13 layer CNN on CIFAR-10 on 50000 images with 4000 labeled examples, and consider weights explored starting half way through training (exact architectural details and training schedules are specified in the Appendix). Figure 2 (middle) shows a histogram of $C(\alpha)$ where the coefficients α_i are sampled from the uniform distribution on the standard simplex (Dirichlet distribution with all parameters equal to one). For all three methods and almost all random convex combinations the value of C is positive, and thus we expect averaging weights to be advantageous. Notably, for the Π model and Mean Teacher the values of C are larger than for standard supervised training, suggesting that we have more to gain by averaging the solutions found by these semi-supervised methods.



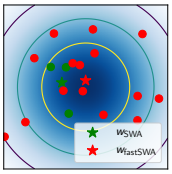


Figure 3: **Left**: Cyclical cosine learning rate schedule and SWA and fast-SWA averaging strategies. **Right**: Illustration of SWA and fast-SWA averaging strategies. fast-SWA averages more points, but the variance of the averaged points is higher.

In Figure 2 (right) we visualize the benefit of averaging weights as a function of the distance between the weights. In particular we sample random pairs of weights w_1, w_2 from the same set in Figure 2 (middle) and evaluate C with $n = 2$, $\alpha_1 = \alpha_2 = 0.5$ against the L_2 distance $\|w_1 - w_2\|$ between the points. We see that we get larger improvement averaging more distant points. We also observe that the the distance between weights along the SGD trajectory is typically higher for the Mean Teacher and Π models than it is for the supervised-only training, and, accordingly, the benefit of averaging these weights is larger. Incidentally, one might expect the EMA training procedure in the Mean Teacher model to increase the distance of weights traversed by SGD relative to the Π model, but this is not the case. Moreover, although the distance is comparable for Mean Teacher and Π models, C is often less for the Mean Teacher, suggesting that weight averaging is likely to have a greater benefit for the Π model.

4 Fast-SWA: Finding Better Solutions by Averaging

In Section 3 we investigated the geometry of solutions obtained by the Mean Teacher and Π models, with two key findings: (1) optima found by these models are flatter but narrower than for fully supervised training; (2) due to the approximate convexity of the SGD training trajectory, averaging weights is likely to be advantageous. Stochastic Weight Averaging (SWA) [10] is a recent training approach that leads to (1) wider optima and better generalization, and (2) is based on averaging weights at different epochs of SGD. Thus, the properties of SWA are wonderfully aligned with improving performance for these consistency-based semi-supervised models.

SWA typically starts from a pre-trained model, and then averages points in weight space traversed by SGD with a constant or cyclical learning rate. In Figure 3 (left), we illustrate the cyclical cosine learning rate schedule. For the first $\ell \leq \ell_0$ epochs the network is pre-trained using the cosine annealing schedule where the learning rate at epoch i is set equal to $\eta(i) = 0.5 \cdot \eta_0(1 + \cos(\pi \cdot i/\ell_0))$. After ℓ epochs, we use a cyclical schedule, repeating the learning rates from epochs $[\ell - c, \ell]$, where c is the cycle length. SWA collects the networks corresponding to the minimum values of the learning rate (shown in green in Figure 3, left) and averages their weights. The model with the averaged weights is then used to make predictions.

Izmailov et al. [10] proposed using cyclical learning rates with small cycle length for SWA. However, as we have seen in Section 3.2 (Figure 2, left) the benefits of averaging are the most prominent when the distance between the averaged points is large. Motivated by this observation, we instead use longer learning rate cycles. Moreover, SWA updates the average of the weights only once per cycle, which translates to many epochs of training in order to obtain enough weights in the average. To overcome this limitation, we propose **fast-SWA**, a modification of SWA that averages networks corresponding to every $k < c$ epochs starting from epoch $\ell - c$. The epochs and learning rates corresponding to the models included in

the fast-SWA average are shown in red in Figure 3 (left). Updating the average multiple times during a single cycle, fast-SWA converges substantially faster than the original SWA for long cycle learning rate schedules. We show empirically in Section 5 that fast-SWA can achieve similar performance to SWA with far fewer epochs.

Notice that most of the models included in the fast-SWA average (shown in red in Figure 3, left) have higher errors than those included in the SWA average (shown in green in Figure 3, left) since they are obtained when the learning rate is high. It is our contention that including more models in the fast-SWA weight average can more than compensate for the larger errors of the individual models. As discussed in Mandt et al. [18], under certain assumptions SGD samples from a Gaussian distribution centered at the optimum of the loss w_0 with covariance proportional to the learning rate. Suppose then that we have n weights sampled at learning rate η_1, $w_i^{(1)} \overset{iid}{\sim} \mathcal{N}(w_0, \eta_1\Sigma)$ and m weights sampled with the higher learning rate η_2, $w_j^{(2)} \overset{iid}{\sim} \mathcal{N}(w_0, \eta_2\Sigma)$. For the SWA estimator $\hat{w}_{\text{SWA}} = \frac{1}{n}\sum_i w_i^{(1)}$, $\mathbb{E}[\|\hat{w}_{\text{SWA}} - w_0\|^2] = \text{tr}(\text{Cov}(\hat{w}_{\text{SWA}})) = \frac{\eta_1}{n}\text{tr}(\Sigma)$. But if we include the high variance points in the average, as in fast-SWA, $\hat{w}_{\text{fSWA}} = \frac{1}{n+m}\big(\sum_i w_i^{(1)} + \sum_j w_j^{(2)}\big)$, then $\mathbb{E}[\|\hat{w}_{\text{fSWA}} - w_0\|^2] = \frac{n\eta_1+m\eta_2}{(n+m)^2}\text{tr}(\Sigma)$. If $\frac{n\eta_1+m\eta_2}{(n+m)^2} < \frac{\eta_1}{n}$ then including the high learning rate points decreases the MSE of the estimator for $m > n\big(\frac{\eta_2}{\eta_1} - 2\big)$.

In our experiments of Section 5, we indeed find that fast-SWA substantially improves the predictive performance of the Π and Mean Teacher models.

5 Experiments

Motivated by our geometric observations, we proposed (1) to improve the Π and Mean Teacher performance using stochastic weight averaging (SWA); (2) fast-SWA, which improves the convergence speed of SWA by averaging multiple weights traversed by SGD on each cycle of a cyclical learning rate schedule.

In this section, we evaluate the Π and MT models with SWA and fast-SWA (Section 4) on CIFAR-10 and CIFAR-100 with varying numbers of labeled examples, compared to conventional SGD training. We show that both fast-SWA and SWA significantly improve the performance of the Π and Mean Teacher models, as we expect from our observations in Sections 3 and 4. In fact, in many cases fast-SWA improves on the best known results in the semi-supervised literature. We also demonstrate that the proposed fast-SWA is able to achieve results comparable to SWA much more quickly than SWA. We also evaluate fast-SWA applied to a consistency-based domain adaptation model [5], closely related to the Mean Teacher model, for adapting CIFAR-10 to STL. We improve the best reported test error rate for this task from 19.9% to 17.1%.

We discuss the experimental setup in Section 5.1. We provide the results for CIFAR-10 and CIFAR-100 datasets in Section 5.2 and 5.3 where we compare the base models with SWA and fast-SWA. We summarize our results in comparison to the best previous results in Section 5.4. We show several additional results and detailed comparisons in Appendix A.1. In Appendix A.5 we also perform an analysis analogous to Section 3, showing that SWA further increases the width of solutions discovered by consistency enforcing models.

5.1 Setup

We evaluate the weight averaging methods SWA and fast-SWA on different network architectures and learning rate schedules, where we are able to improve on the base models in all settings. In particular, we consider a 13-layer CNN and a 12-block (26-layer) Residual

Network [9] with Shake-Shake regularization [6] (which we refer to simply as *CNN* and *Shake-Shake*, respectively). See Section A.6 for details on the architectures. For training all methods we use the stochastic gradient descent optimizer with the cosine annealing learning rate described in Section 4. We use two learning rate schedules, the *short schedule* with $\ell = 180, \ell_0 = 210, c = 30$ similar to the experiments in Tarvainen and Valpola [28] and the *long schedule* with $\ell = 1500, \ell_0 = 1800, c = 200$. We note that the long schedule improves the performance of the base models compared to the short schedule; however, fast-SWA and SWA can still further improve the results. See Section A.7 of the Appendix for more details on other hyperparameters. We repeat each CNN experiment 3 times with different random seeds to estimate the standard deviations for the results in the Appendix.

5.2 CIFAR-10

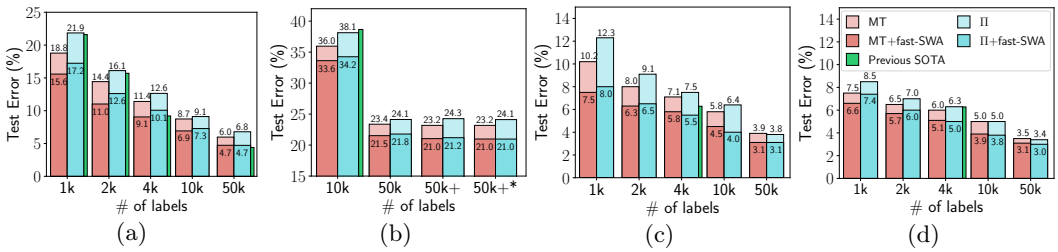


Figure 4: Prediction errors of Π and MT models with and without fast-SWA. **(a)** CIFAR-10 with CNN **(b)** CIFAR-100 with CNN. $50k+$ and $50k+^*$ correspond to $50k$+$500k$ and $50k$+$237k^*$ settings **(c)** CIFAR-10 with ResNet + Shake-Shake using the short schedule **(d)** CIFAR-10 with ResNet + Shake-Shake using the long schedule.

We evaluate the proposed fast-SWA method using the Π and Mean Teacher models on the CIFAR-10 dataset [14]. We use $50k$ images for training with $1k$, $2k$, $4k$, $10k$ and $50k$ labels and report the errors on the test set ($10k$ images). We visualize the results for the CNN and Shake-Shake architectures in Figures 4a, 4c, and 4d. For all quantities of labeled data, fast-SWA substantially improves test accuracy in both architectures. Additionally, in Tables 3, 5 of the Appendix we provide a thorough comparison of different averaging strategies as well as results for VAT [19], TE [15], and other baselines.

In Figure 5 (left), we visualize the test error as a function of iteration using the CNN. We observe that when the cyclical learning rate starts after epoch $\ell = 180$, the base models drop in performance due to the sudden increase in learning rate (see Figure 3 left) However, fast-SWA continues to improve while collecting the weights corresponding to high learning rates for averaging. In general, we also find that the cyclical learning rate improves the base models beyond the usual cosine annealing schedule and pushes the performance of fast-SWA as training progresses. Compared to SWA, we also observe that fast-SWA converges substantially faster, for instance, reducing the error to 10.5% at epoch 200 while SWA attains similar error at epoch 350 for CIFAR-10 $4k$ labels (Figure 5 left). We provide additional plots in Section A.1 for the convergence trends of the Π and Mean Teacher models in all label settings, where we observe similar trends that fast-SWA results in faster error reduction.

We also find that the performance gains of fast-SWA over base models are higher for the Π model compared to the MT model, which is consistent with the convexity observation in Section 3.2 and Figure 2. In the previous evaluations [see e.g. 20, 28], the Π model was shown to be inferior to the Mean Teacher model. However, with weight averaging, fast-SWA reduces the gap between Π and MT performance. Surprisingly, we find that the Π model can outperform Mean Teacher after applying fast-SWA with moderate to large numbers

of labeled points. In particular, the Π+fast-SWA model outperforms MT+fast-SWA on CIFAR-10 with $4k$, $10k$, and $50k$ labeled data points for the Shake-Shake architecture.

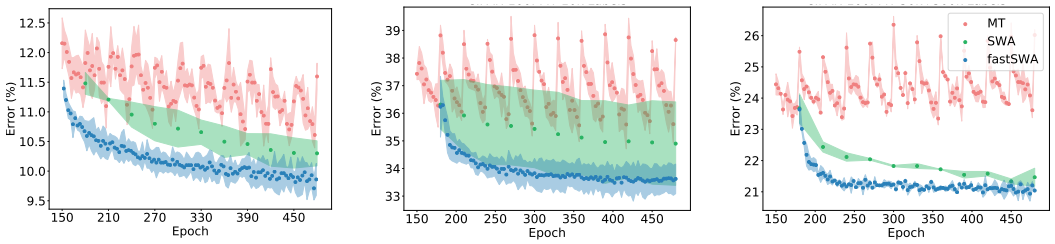


Figure 5: Prediction errors of base models and their weight averages (fast-SWA and SWA) for CNN on **(left)** CIFAR-10 with $4k$ labels, **(middle)** CIFAR-100 with $10k$ labels, and **(right)** CIFAR-100 $50k$ labels and extra $500k$ unlabeled data from Tiny Images [29].

5.3 CIFAR-100 and Extra Unlabeled Data

We evaluate the Π and Mean Teacher models with fast-SWA on CIFAR-100. We train our models using 50000 images with $10k$ and $50k$ labels using the 13-layer CNN. Additionally, we analyze the effect of using the Tiny Images dataset [29] as an additional source of unlabeled data.

Tiny Images consists of 80 million images, mostly unlabeled, and contain CIFAR-100 as a subset. Following Laine and Aila [15], we use two settings of unlabeled data, $50k$+$500k$ and $50k$+$237k^*$ where the $50k$ images corresponds to CIFAR-100 images from the training set and the $+500k$ or $+237k^*$ images corresponds to additional $500k$ or $237k$ images from the Tiny Images dataset. For the $237k^*$ setting, we select only the images that belong to the classes in CIFAR-100, corresponding to 237203 images. However, for the $500k$ setting, we use a random set of $500k$ images whose classes can be different from CIFAR-100's. We visualize the results in Figure 4b, where we again observe that fast-SWA substantially improves performance for every configuration of the number of labeled and unlabeled data. In Figure 5 (middle, right) we show the errors of Mean Teacher, SWA and fast-SWA as a function of iteration on CIFAR-100 for the $10k$ and $50k$+$500k$ label settings. Similar to the CIFAR-10 experiments, we observe that fast-SWA reduces the errors substantially faster than SWA.

We provide detailed experimental results in Table 4 of the Appendix and include preliminary results using the Shake-Shake architecture in Table 6, Section A.1.

5.4 Advancing State-of-the-Art

We have shown that fast-SWA can significantly improve the performance of both the Π and Mean Teacher models. We provide a summary comparing our results with the previous best results in the literature in Table 1, using the 13-layer CNN and the Shake-Shake architecture that had been applied previously. Further results are in the Appendix.

5.5 Domain Adaptation

Domain adaptation problems involve learning using a source domain X_s equipped with labels Y_s and performing classification on the target domain X_t while having no access to the target labels at training time. A recent model by French et al. [5] applies the consistency enforcing principle for domain adaptation and achieves state-of-the-art results on many datasets. We apply fast-SWA to the model adapting from CIFAR-10 to STL. We report the results in

Table 1: Test errors against current state-of-the-art semi-supervised results. The previous best numbers are obtained from [28][1], [21][2], [15][3]. CNN denotes performance on the benchmark 13-layer CNN (see A.6). Rows marked † use the Shake-Shake architecture. The result marked ‡ are from Π + fast-SWA, where the rest are based on MT + fast-SWA. The settings $50k$+$500k$ and $50k$+$237k^*$ use additional $500k$ and $237k$ unlabeled data from the Tiny Images dataset [29] where * denotes that we use only the images that correspond to CIFAR-100 classes.

Dataset	CIFAR-10			CIFAR-100		
No. of Images	50k	50k	50k	50k	50k+500k	50k+237k*
No. of Labels	1k	2k	4k	10k	50k	50k
Previous Best CNN	21.55[1]	15.73[1]	9.22[2]	38.65[3]	23.62[3]	23.79[3]
Ours CNN	15.58	11.02	9.05	33.62	21.04	20.98
Previous Best†			6.28[1]			
Ours†	6.6	5.7	5.0‡	28.0	19.3	17.7

Table 2. Again we see that fast-SWA provides large gains in performance improving the best result reported in the literature. We provide experimental details in Section A.8.

Table 2: Domain Adaptation from CIFAR-10 to STL.

Method	VADA [26]	SE [5]	SE + SWA
Test Error	20.0	19.9	**17.1**

6 Discussion

Semi-supervised learning is crucial for reducing the dependency of deep learning on large labeled data. Recently, there have been great advances in semi-supervised learning, with consistency regularization models achieving the best known results. By analyzing the geometry of the training objectives for two of the most successful models in this class, the Π and Mean Teacher models, we proposed to perform stochastic weight averaging (SWA) for training — advancing the best known semi-supervised results on consequential benchmarks. Moreover, we further improved convergence by developing fast-SWA, which was again inspired by geometric considerations.

We have also shown that fast-SWA has promise to improve results in domain adaptation, which we view as an exciting area for future research. While we have focused here on the popular Mean Teacher and Π models, we also believe that fast-SWA will be useful for improving other consistency based semi-supervised learning methods, as well as a wide range of other models, including standard fully-supervised models, in future work. We have also shown that the relatively simple Π model can achieve state-of-the-art-results if we use fast-SWA, which was inspired by investigating the geometry of its training objective.

Analyzing the geometry of a training objective in general provides a powerful mechanism for understanding model behavior and improving predictive performance. There has recently been great interest in relating optima width to generalization [25, 11, 10]. But width is not the only consideration: we have also found that while the Π and Mean Teacher models provide locally flat objectives, these loss functions are often more narrow than for fully supervised objectives. There are also potentially many different metrics for understanding loss function geometry. Here we have considered movement in random and adversarial directions. Exciting future work includes devising new metrics for understanding geometric properties of training objectives, and further investigating how these geometric properties relate to model performance.

References

[1] H. Avron and S. Toledo. Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix. *Journal of the ACM*, 58(2):1–34, Apr. 2011. ISSN 00045411. doi: 10.1145/1944345.1944349.

[2] P. Chaudhari, A. Choromanska, S. Soatto, Y. LeCun, C. Baldassi, C. Borgs, J. Chayes, L. Sagun, and R. Zecchina. Entropy-SGD: Biasing Gradient Descent Into Wide Valleys. *arXiv:1611.01838 [cs, stat]*, Nov. 2016. arXiv: 1611.01838.

[3] Z. Dai, Z. Yang, F. Yang, W. W. Cohen, and R. R. Salakhutdinov. Good semi-supervised learning that requires a bad gan. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6510–6520. Curran Associates, Inc., 2017.

[4] L. Dinh, R. Pascanu, S. Bengio, and Y. Bengio. Sharp Minima Can Generalize For Deep Nets. *arXiv:1703.04933 [cs]*, Mar. 2017.

[5] G. French, M. Mackiewicz, and M. Fisher. Self-ensembling for visual domain adaptation. In *International Conference on Learning Representations*, 2018.

[6] X. Gastaldi. Shake-shake regularization. *CoRR*, abs/1705.07485, 2017.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[8] I. Goodfellow, O. Vinyals, and A. Saxe. Qualitatively characterizing neural network optimization problems. *International Conference on Learning Representations*, 2015.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[10] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson. Averaging weights leads to wider optima and better generalization. 2018.

[11] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *ICLR*, 2017.

[12] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[13] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3581–3589, 2014.

[14] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. page 60.

[15] S. Laine and T. Aila. Temporal Ensembling for Semi-Supervised Learning. *arXiv:1610.02242 [cs]*, Oct. 2016.

[16] S. Laine and T. Aila. Temporal ensembling for semi-supervised learning. *International Conference on Learning Representations*, 2017.

[17] I. Loshchilov and F. Hutter. SGDR: stochastic gradient descent with restarts. *CoRR*, abs/1608.03983, 2016.

[18] S. Mandt, M. D. Hoffman, and D. M. Blei. Stochastic gradient descent as approximate bayesian inference. *arXiv preprint arXiv:1704.04289*, 2017.

[19] T. Miyato, S. Maeda, M. Koyama, and S. Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *CoRR*, abs/1704.03976, 2017.

[20] A. Oliver, A. Odena, C. Raffel, E. D. Cubuk, and I. J. Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. *ICLR Workshop*, 2018.

[21] S. Park, J.-K. Park, S.-J. Shin, and I.-C. Moon. Adversarial Dropout for Supervised and Semi-supervised Learning. *arXiv:1707.03631 [cs]*, July 2017. arXiv: 1707.03631.

[22] Y. Saatci and A. G. Wilson. Bayesian gan. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3622–3631. Curran Associates, Inc., 2017.

[23] M. Sajjadi, M. Javanmardi, and T. Tasdizen. Regularization With Stochastic Transformations and Perturbations for Deep Semi-Supervised Learning. *arXiv:1606.04586 [cs]*, June 2016. arXiv: 1606.04586.

[24] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, and X. Chen. Improved techniques for training gans. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc., 2016.

[25] J. Schmidhuber and S. Hochreiter. Flat minima. *Neural Computation*, 1997.

[26] R. Shu, H. Bui, H. Narui, and S. Ermon. A DIRT-t approach to unsupervised domain adaptation. In *International Conference on Learning Representations*, 2018.

[27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[28] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 1195–1204, 2017.

[29] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(11):1958–1970, 2008.

[30] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. page 8.

A Appendix

A.1 Detailed Results

In this section we report detailed results for the Π and Mean Teacher models and various baselines on CIFAR-10 and CIFAR-100 using the 13-layer CNN and Shake-Shake.

The results using the 13-layer CNN are summarized in Table 3 and 4 for CIFAR-10 and CIFAR-100 respectively. Table 5 and 6 summarizes the results using Shake-Shake on CIFAR-10 and CIFAR-100. In the tables Π EMA is the same method as Π , where instead of SWA we apply Exponential Moving Averaging (EMA) for the student weights. We show that simply performing EMA typically results in a small again over the student network without using it as a teacher network.

Figure 6 and 7 show the performance trend of the Π and Mean Teacher models over the number of epochs for CIFAR-10 and CIFAR-100 respectively.

Table 3: CIFAR-10 semi-supervised errors on test set with a 13-layer CNN. The epoch numbers are reported in parenthesis.

Number of labels	1000	2000	4000	10000	50000
TE [15]	-	-	12.16 ± 0.31		5.60 ± 0.15
Supervised-only [28]	46.43 ± 1.21	33.94 ± 0.73	20.66 ± 0.57		5.82 ± 0.15
Π [28]	27.36 ± 1.20	18.02 ± 0.60	13.20 ± 0.27		6.06 ± 0.15
MT [28]	21.55 ± 1.48	15.73 ± 0.31	12.31 ± 0.28		5.94 ± 0.15
VAT + EntMin [19]			10.55		
VAdD [21]			9.22 ± 0.10		**4.40 ± 0.12**
MT	18.78 ± 0.31	14.43 ± 0.20	11.41 ± 0.27	8.74 ± 0.30	5.98 ± 0.21
MT + fast-SWA (180)	18.19 ± 0.38	13.46 ± 0.30	10.67 ± 0.18	8.06 ± 0.12	5.90 ± 0.03
MT + fast-SWA (240)	17.81 ± 0.37	13.00 ± 0.31	10.34 ± 0.14	7.73 ± 0.10	5.55 ± 0.03
MT + SWA (240)	18.38 ± 0.29	13.86 ± 0.64	10.95 ± 0.21	8.36 ± 0.50	5.75 ± 0.29
MT + fast-SWA (480)	16.84 ± 0.62	12.24 ± 0.31	9.86 ± 0.27	7.39 ± 0.14	5.14 ± 0.07
MT + SWA (480)	17.48 ± 0.13	13.09 ± 0.80	10.30 ± 0.21	7.78 ± 0.49	5.31 ± 0.43
MT + fast-SWA (1200)	**15.58 ± 0.12**	**11.02 ± 0.23**	**9.05 ± 0.21**	**6.92 ± 0.07**	**4.73 ± 0.18**
MT + SWA (1200)	15.59 ± 0.77	11.42 ± 0.33	9.38 ± 0.28	7.04 ± 0.11	5.11 ± 0.35
Π	21.85 ± 0.69	16.10 ± 0.51	12.64 ± 0.11	9.11 ± 0.21	6.79 ± 0.22
Π EMA	21.70 ± 0.57	15.83 ± 0.55	12.52 ± 0.16	9.06 ± 0.15	6.66 ± 0.20
Π + fast-SWA (180)	20.79 ± 0.38	15.12 ± 0.44	11.91 ± 0.06	8.83 ± 0.32	6.42 ± 0.09
Π + fast-SWA (240)	20.04 ± 0.41	14.77 ± 0.15	11.61 ± 0.06	8.45 ± 0.28	6.14 ± 0.11
Π + SWA (240)	21.37 ± 0.64	15.38 ± 0.85	12.05 ± 0.40	8.58 ± 0.41	6.36 ± 0.55
Π + fast-SWA (480)	19.11 ± 0.29	13.88 ± 0.30	10.91 ± 0.15	7.91 ± 0.21	5.53 ± 0.07
Π + SWA (480)	20.06 ± 0.64	14.53 ± 0.81	11.35 ± 0.42	8.04 ± 0.37	5.77 ± 0.51
Π + fast-SWA (1200)	**17.23 ± 0.34**	**12.61 ± 0.18**	**10.07 ± 0.27**	**7.28 ± 0.23**	**4.72 ± 0.04**
Π + SWA (1200)	17.70 ± 0.25	12.59 ± 0.29	10.73 ± 0.39	7.13 ± 0.23	4.99 ± 0.41

Table 4: CIFAR-100 semi-supervised errors on test set. All models are trained on a 13-layer CNN. The epoch numbers are reported in parenthesis.

Number of labels	10k	50k	50k + 500k	50k + 237k*
Supervised-only [15]	44.56 ± 0.30	26.42 ± 0.17		
Π model [15]	39.19 ± 0.54	26.32 ± 0.04	25.79 ± 0.17	25.43 ± 0.17
Temporal Ensembling [15]	38.65 ± 0.51	26.30 ± 0.15	23.62 ± 0.17	23.79 ± 0.17
MT (180)	35.96 ± 0.77	23.37 ± 0.16	23.18 ± 0.06	23.18 ± 0.24
MT + fast-SWA (180)	34.54 ± 0.48	21.93 ± 0.16	21.04 ± 0.16	21.09 ± 0.12
MT + SWA (240)	35.59 ± 1.45	23.17 ± 0.86	22.00 ± 0.23	21.59 ± 0.22
MT + fast-SWA (240)	34.10 ± 0.31	21.84 ± 0.12	21.16 ± 0.21	21.07 ± 0.21
MT + SWA (1200)	34.90 ± 1.51	22.58 ± 0.79	21.47 ± 0.29	21.27 ± 0.09
MT + fast-SWA (1200)	**33.62 ± 0.54**	**21.52 ± 0.12**	**21.04 ± 0.04**	**20.98 ± 0.36**
Π (180)	38.13 ± 0.52	24.13 ± 0.20	24.26 ± 0.15	24.10 ± 0.07
Π + fast-SWA (180)	35.59 ± 0.62	22.08 ± 0.21	21.40 ± 0.19	21.28 ± 0.20
Π + SWA (240)	36.89 ± 1.51	23.23 ± 0.70	22.17 ± 0.19	21.65 ± 0.13
Π + fast-SWA (240)	35.14 ± 0.71	22.00 ± 0.21	21.29 ± 0.27	21.22 ± 0.04
Π + SWA (1200)	35.35 ± 1.15	22.53 ± 0.64	21.53 ± 0.13	21.26 ± 0.34
Π + fast-SWA (1200)	**34.25 ± 0.16**	**21.78 ± 0.05**	**21.19 ± 0.05**	**20.97 ± 0.08**

Table 5: CIFAR-10 semi-supervised errors on test set. All models use Shake-Shake Regularization [6] + ResNet-26 [9].

Number of labels	1000	2000	4000	10000	50000
Short Schedule ($\ell = 180$)					
MT† [28]			6.28		
MT (180)	10.2	8.0	7.1	5.8	3.9
MT + SWA (240)	9.7	7.7	6.2	4.9	3.4
MT + fast-SWA (240)	9.6	7.4	6.2	4.9	3.2
MT + SWA (1200)	7.6	6.4	5.8	4.6	**3.1**
MT + fast-SWA (1200)	**7.5**	**6.3**	5.8	4.5	**3.1**
Π (180)	12.3	9.1	7.5	6.4	3.8
Π + SWA (240)	11.0	8.3	6.7	5.5	3.3
Π + fast-SWA (240)	11.2	8.2	6.7	5.5	3.3
Π + SWA (1200)	8.2	6.7	5.7	4.2	**3.1**
Π + fast-SWA (1200)	8.0	6.5	**5.5**	**4.0**	**3.1**
Long Schedule ($\ell = 1500$)					
Supervised-only [6]					**2.86**
MT (1500)	7.5	6.5	6.0	5.0	3.5
MT + fast-SWA (1700)	6.4	5.8	5.2	3.8	3.4
MT + SWA (1700)	6.9	5.9	5.5	4.2	3.2
MT + fast-SWA (3500)	**6.6**	**5.7**	5.1	3.9	3.1
MT + SWA (3500)	6.7	5.8	5.2	3.9	3.1
Π (1500)	8.5	7.0	6.3	5.0	3.4
Π + fast-SWA (1700)	7.5	6.2	5.2	4.0	3.1
Π + SWA (1700)	7.8	6.4	5.6	4.4	3.2
Π + fast-SWA (3500)	7.4	6.0	**5.0**	**3.8**	3.0
Π + SWA (3500)	7.9	6.2	5.1	4.0	3.0

Table 6: CIFAR-100 semi-supervised errors on test set. Ours models use Shake-Shake Regularization [6] + ResNet-26 [9].

Number of labels	10k	50k	50k + 500k	50k + 237k*
TE (CNN) [15]	38.65 ± 0.51	26.30 ± 0.15	23.62 ± 0.17	23.79 ± 0.17
Short Schedule ($\ell = 180$)				
MT (180)	29.4	19.5	21.9	19.0
MT + fast-SWA (180)	28.9	19.3	19.7	18.3
MT + SWA (240)	28.4	18.8	19.9	17.9
MT + fast-SWA (240)	28.1	18.8	19.5	17.9
MT + SWA (300)	28.1	18.5	**18.9**	**17.5**
MT + fast-SWA (300)	**28.0**	**18.4**	19.3	17.7

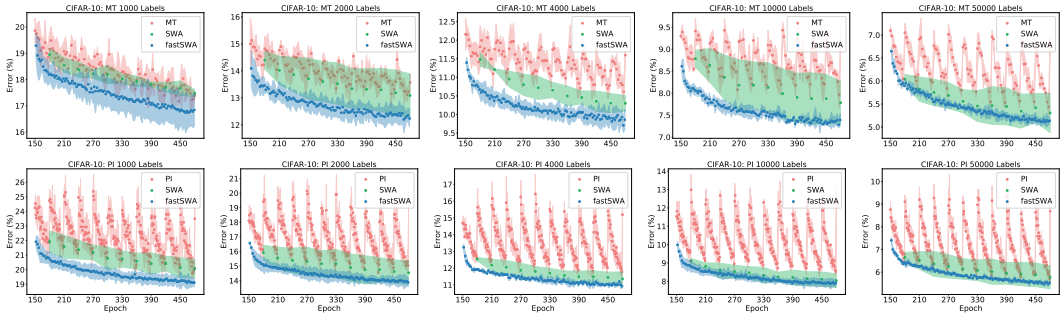


Figure 6: Test errors as a function of training epoch for baseline models, SWA and fast-SWA on CIFAR-10 trained using $1k$, $2k$, $4k$, and $10k$ labels for **(top)** the MT model **(bottom)** the Π model. All models are trained using the 13-layer CNN.

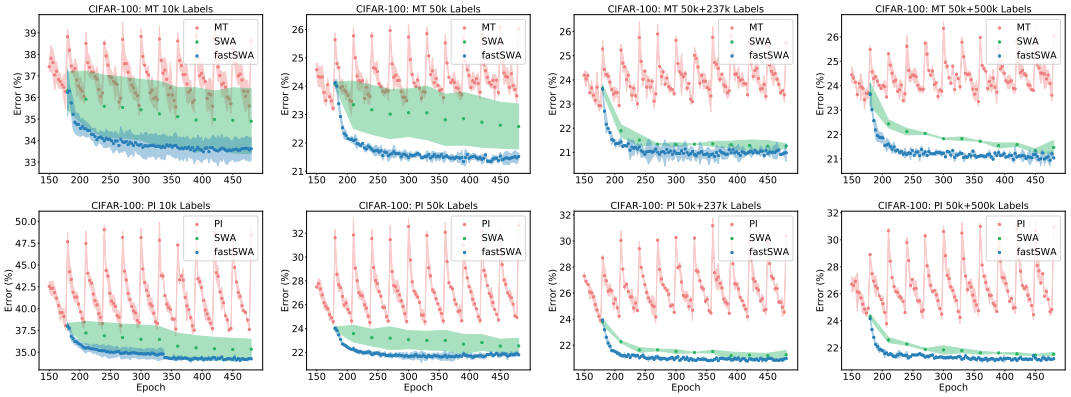
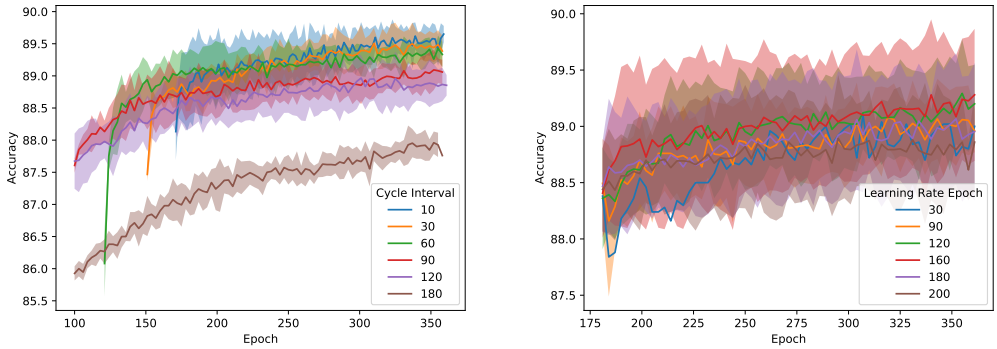


Figure 7: Test errors versus training epoch for baseline models, SWA and fast-SWA on CIFAR-100 trained using $10k$, $50k$, $50k$+$500k$, and $50k$+$237k^*$ labels for **(top)** the MT model **(bottom)** the Π model. All models are trained using the 13-layer CNN.

A.2 Effect of Learning Rate Schedules

The only hyperparameter for the fast-SWA setting is the cycle interval c. We demonstrate in Figure 8a that fast-SWA's performance is not sensitive to c on a wide range of c values. We also demonstrate the performance of constant learning schedule. fast-SWA with cyclical learning rates often converge faster due to a variety of weight points collected.



(a) (b)

Figure 8: (a) Effects of Cycle Interval c (b) fast-SWA with constant learning rate. The "learning rate epoch" corresponds to the epoch at which the learning rate is evaluated (Figure 3, left). We use this learning rate as a constant rate for epoch $i \geq \ell$. The model is trained on $45k$ images from the original $50k$ and evaluated on $5k$ validation split.

A.3 Consistency Loss Encourages Flat Minima

The notion of wide and sharp optima has seen a lot of attention recently [see e.g. 25, 11, 10] and has been theorized to relate to generalization performance. As pointed out in Dinh et al. [4] and Chaudhari et al. [2], the local information about sharpness is encoded into the eigenvalues of H, the Hessian of the loss. Specifically the trace and the spectral norm of the Hessian give two ways of measuring sharpness. As shown in the A.4, for the MSE loss, $\mathbb{E}_x[\text{tr}(H)]$ is the sum of two terms, one of which is $\mathbb{E}_x[\|J_w\|_F^2]$, the expected Frobenius norm of the Jacobian. In this section we show that the consistency loss in a simplified Π model directly penalizes $\mathbb{E}_x[\|J_w\|_F^2]$, and thus encourages flat minima in the sense of $\mathbb{E}_x[\text{tr}(H)]$.

Consider a simple version of the Π model, where we only apply small additive perturbations to the student weights: $w' = w + \epsilon z,\ z \sim \mathcal{N}(0, I)$ with $\epsilon \ll 1$, and the teacher weights are unchanged: $w'' = w$.[1] Then the consistency loss ℓ_{cons} defined in (1) can be written as $\ell_{cons}(w, x) = \|f(w + \epsilon z, x) - f(w, x)\|^2$. Taylor expanding in ϵ, we obtain

$$\ell_{cons}(w, x) = \epsilon^2 z^T J_w^T J_w z + O(\epsilon^4) \tag{3}$$

where $J_w(w, x)$ is the Jacobian of f with respect to the weights for a fixed input x. We can now recognize this term as a one sample stochastic trace estimator for $\epsilon^2 \text{tr}(J_w^T J_w)$ with a Gaussian probe variable z; see Avron and Toledo [1] for derivations and guarantees on stochastic trace estimators. Therefore, the average consistency loss $L_{cons}/N = \frac{1}{N}\sum_{i=1}^{N} \ell_{cons}(w, x_i)$ approximates $\epsilon^2 \mathbb{E}_x[\text{tr}(J_w^T J_w)] = \epsilon^2 \mathbb{E}_x[\|J_w\|_F^2]$.

Thus, for small additive weight perturbations, L_{cons} term of the Π model approximates the average Frobenius norm of the Jacobian. As discussed in the beginning of this section, penalizing this norm encourages model robustness.

Similarly for small additive perturbations to the input image x, the consistency term corresponds to the Frobenius norm of J_x, the Jacobian with respect to the data input. This kind of regularization corresponds exactly to weight decay, also known as L_2 regularization for linear models $f(x) = Wx$, since $J_x = W$, and $\|W\|_F^2 = \|\text{vec}(W)\|_2^2$. This regularization is also related to the graph based regularization in [30] that approximates $E_x[\|\nabla_x f\|^2]$ for a univariate function using the graph Laplacian.

[1] This assumption can be relaxed to $w'' = w + \epsilon z_2 \quad z_2 \sim \mathcal{N}(0, I)$ without changing the results of the analysis since $\epsilon(z_1 - z_2) = 2\epsilon\bar{z}$ with $\bar{z} \sim \mathcal{N}(0, I)$

A.4 Relationship Between $\mathbb{E}_x[\|J_w\|_F^2]$ and Broad Optima

In the following analysis we will show that smaller $\mathbb{E}_x[\|J_w\|_F^2]$, implies broader optima. To keep things simple, we focus on the MSE loss, but in principle a similar argument should apply for the Cross Entropy and the Error rate. For a single data point x and one hot vector y with k classes, the hessian of $\ell_{MSE}(w) = \|f(x,w) - y\|^2$ is

$$H(w,x,y) = \nabla^2 \ell_{MSE}(w) = J_w^T J_w + \sum_{i=1}^{k} (\nabla^2 f_i)(f_i(x) - y_i),$$

$$\mathrm{tr}(H) = \|J_w\|_F^2 + \underbrace{\sum_{i=1}^{k} \mathrm{tr}(\nabla^2 f_i)(f_i(x) - y_i)}_{\alpha(x,y)}$$

So the $\mathrm{tr}(H)$ is the sum of two terms, $\|J_w\|_F^2$ and α. As the solution improves, the relative importance of α goes down. Tying this back to random ray sharpness, consider the expected MSE loss, or risk, $R_{\mathrm{MSE}}(w) = \mathbb{E}_{(x,y)}\|f(x,w) - y\|^2$ along random rays. Let d be a random vector sampled from the unit sphere and s is the distance along the random ray. Evaluating the risk on a random ray, and Taylor expanding in s we have

$$R_{\mathrm{MSE}}(w + sd) = R_{\mathrm{MSE}}(w) + sd^T \mathbb{E}_{(x,y)}[J_w^T(f - y)] + (1/2)s^2 d^T \mathbb{E}_{(x,y)}[H]d + O(s^3)$$

Since d is from the unit sphere, $\mathbb{E}[d] = 0$ and $\mathbb{E}[dd^T] = I/p$ where p is the dimension. Averaging over the rays, $d \sim \mathrm{Unif}(S^{p-1})$, we have

$$\mathbb{E}_d[R_{\mathrm{MSE}}(w+sd)] - R_{\mathrm{MSE}}(w) = \frac{s^2}{2p}\mathbb{E}_x[\mathrm{tr}(H)] + O(s^4) = \frac{s^2}{2p}\mathbb{E}_x[\|J_w\|_F^2] + \frac{s^2}{2p}\mathbb{E}_{(x,y)}[\alpha(x,y)] + O(s^4)$$

All of the odd terms vanish because of the reflection symmetry of the unit sphere. This means that locally, the sharpness of the optima (as measured by random rays) can be lowered by decreasing $\mathbb{E}_x[\|J_w\|_F^2]$.

A.5 Analysis

In this section we analyze the effect of applying fast-SWA on the geometry of solutions obtained by Π model and Mean Teacher. We follow the evaluation framework of section 3.1. In Figure 9 we plot the test error as a function of distance along random and adversarial rays starting at conventional and fast-SWA solutions. We align each of the error curves in Figure 9 (left) and (middle) at a distance of $s = 0$, to directly compare their shape.

We find that SWA applied to MT and the Π model increases the width of the solutions along both random and adversarial rays. In general the increase in width is less prominent than that reported by Izmailov et al. [10] for supervised learning. To further analyze the geometry of SWA and conventional solutions, we evaluate the train and test error along the line $\phi(t) = t \cdot w_{\mathrm{SWA}} + (1-t) \cdot w_{\mathrm{SGD}}$ connecting the SWA solution w_{SWA} for Π model and the conventional Π model solution w_{SGD}. We show the results in Figure 9 (right). Along this direction the fast-SWA solution exhibits behavior similar to that observed in Izmailov et al. [10] for supervised case. fast-SWA solution is more centered than SGD solution in terms of train error, and achieves the minimum of test error along the specified line. Like in the fully-supervised case, fast-SWA solutions are more centered in the region of weight space corresponding to networks with low test error.

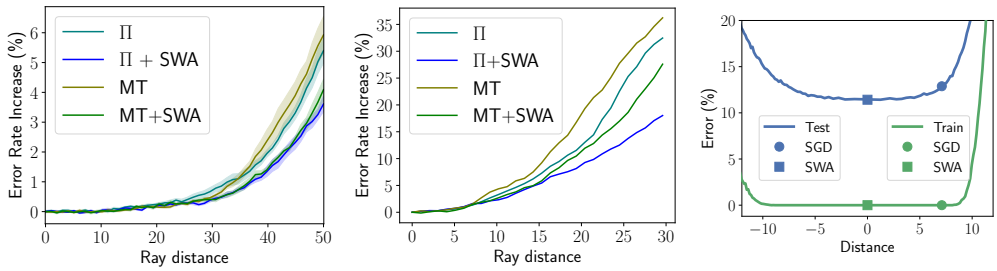


Figure 9: **Left:** Test accuracy as a function of distance along random rays for the Π model and Mean Teacher with and without fast-SWA. **Middle:** Test accuracy as a function of distance along the adversarial ray for the Π model and Mean Teacher with and without fast-SWA. **Right**: Train and test accuracy along the line connecting the conventional Π model solution and fast-SWA solution. All plots are generated using $4k$ labeled examples and $46k$ unlabeled examples.

A.6 Network Architectures

In the experiments we use two DNN architectures – 13 layer CNN and Shake-Shake. The architecture of 13-layer CNN is described in Table 7. It closely follows the architecture used in [16, 19, 28]. We re-implement it in PyTorch and removed the Gaussian input noise, since we found having no such noise improves generalization. For Shake-Shake we use 26-2x96d Shake-Shake regularized architecture of Gastaldi [6] with 12 residual blocks.

Table 7: A 13-layer convolutional neural networks for the CNN experiments (CIFAR-10 and CIFAR-100) in Section 5.2 and 5.3. Note that the difference from the architecture used in Tarvainen and Valpola [28] is that we removed a Gaussian noise layer after the horizontal flip.

Layer	**Hyperparameters**
Input	32×32 RGB image
Translation	Randomly $\{\Delta x, \Delta y\} \sim [-2, 2]$
Horizontal flip	Randomly $p = 0.5$
Convolutional	128 filters, 3×3, *same* padding
Convolutional	128 filters, 3×3, *same* padding
Convolutional	128 filters, 3×3, *same* padding
Pooling	Maxpool 2×2
Dropout	$p = 0.5$
Convolutional	256 filters, 3×3, *same* padding
Convolutional	256 filters, 3×3, *same* padding
Convolutional	256 filters, 3×3, *same* padding
Pooling	Maxpool 2×2
Dropout	$p = 0.5$
Convolutional	512 filters, 3×3, *valid* padding
Convolutional	256 filters, 1×1, *same* padding
Convolutional	128 filters, 1×1, *same* padding
Pooling	Average pool ($6 \times 6 \rightarrow 1 \times 1$ pixels)
Softmax	Fully connected $128 \rightarrow 10$

A.7 Hyperparameters

We consider two different schedules. In the *short schedule* we set the cosine half-period $\ell_0 = 210$ and training length $\ell = 180$, following the schedule used in Tarvainen and Valpola [28] in Shake-Shake experiments. For our Shake-Shake experiments we also report results with *long schedule* where we set $\ell = 1800, \ell_0 = 1500$ following Gastaldi [6]. To determine the initial learning rate η_0 and the cycle length c we used a separate validation set of size 5000 taken from the unlabeled data. After determining these values, we added the validation set to the unlabeled data and trained again. We reuse the same values of η_0 and c for all experiments with different numbers of labeled data for both Π model and Mean Teacher for a fixed architecture (13-layer CNN or Shake-Shake). For the short schedule we use cycle length $c = 30$ and average models once every $k = 3$ epochs. For long schedule we use $c = 200$, $k = 20$.

In all experiments we use stochastic gradient descent optimizer with Nesterov momentum [17]. In fast-SWA we average every the weights of the models corresponding to every third epoch. In the Π model, we back-propagate the gradients through the student side only (as opposed to both sides in [15]). For Mean Teacher we use $\alpha = 0.97$ decay rate in the Exponential Moving Average (EMA) of the student's weights. For all other hyper-parameters we reuse the values from Tarvainen and Valpola [28] unless mentioned otherwise.

Like in Tarvainen and Valpola [28], we use $\|\cdot\|^2$ for divergence in the consistency loss. Similarly, we ramp up the consistency cost λ over the first 5 epochs from 0 up to it's maximum value of 100 as done in [28]. We use cosine annealing learning rates with no learning rate ramp up, unlike in the original MT implementation [28]. Note that this is similar to the same hyperparameter settings as in Tarvainen and Valpola [28] for ResNet[2]. We note that we use the exact same hyperparameters for the Π and MT models in each experiment setting. In contrast to the original implementation in Tarvainen and Valpola [28] of CNN experiments, we use SGD instead of Adam.

CIFAR-10 CNN Experiments We use a total batch size of 100 for CNN experiments with a labeled batch size of 50. We use the maximum learning rate $\eta_0 = 0.1$.

CIFAR-10 ResNet + Shake-Shake We use a total batch size of 128 for ResNet experiments with a labeled batch size of 31. We use the maximum learning rate $\eta_0 = 0.05$ for CIFAR-10. This applies for both the short and long schedules.

CIFAR-100 CNN Experiments We use a total batch size of 128 with a labeled batch size of 31 for $10k$ and $50k$ label settings. For the settings $50k$+$500k$ and $50k$+$237k^*$, we use a labeled batch size of 64. We also limit the number of unlabeled images used in each epoch to $100k$ images. We use the maximum learning rate $\eta_0 = 0.1$.

CIFAR-100 ResNet + Shake-Shake We use a total batch size of 128 for ResNet experiments with a labeled batch size of 31 in all label settings. For the settings $50k$+$500k$ and $50k$+$237k^*$, we also limit the number of unlabeled images used in each epoch to $100k$ images. We use the maximum learning rate $\eta_0 = 0.1$. This applies for both the short and long schedules.

[2]We use the public Pytorch code https://github.com/CuriousAI/mean-teacher as our base model for the MT model and modified it for the Π model.

A.8 Domain Adaptation Implementation

We use the public code[3] of French et al. [5] to train the model and apply fast-SWA. While the original implementation uses Adam [12], we use stochastic gradient descent with Nesterov momentum and cosine annealing learning rate with $\ell_0 = 600, \ell = 550, c = 100$ and $k = 100$. We use the maximum learning rate $\eta_0 = 0.1$ and momentum 0.9 with weight decay of scale 2×10^{-4}. We use the data augmentation setting MT+CF+TFA in Table 1 of French et al. [5] and apply fast-SWA. The result reported is from epoch 4000.

[3] https://github.com/Britefury/self-ensemble-visual-domain-adapt.git